COSMOGRAPHIA-SPICE USER'S GUIDE

Prepared by JPL Summer Intern Students

Michelle Park
and
Farhan Alam

July 22, 2014

Revised by
C. Acton
and
B. Semenov

Version 8.0
August 14, 2015

Table of Contents

I. INTRODUCTION

Cosmographia is a visualization program rendering the solar system and its bodies in 3D to create a freely navigable map of the solar system. The program allows manipulation of time and observer position. It can use SPICE data to visualize trajectory, orientation, and sensors flown on and observations taken by interplanetary spacecraft, to support scientific or engineering analysis, and perhaps even public outreach.

This User's Guide illustrates how to produce catalog files in the JavaScript Object Notation (JSON) format necessary to use SPICE data in Cosmographia. It is intended to be used in conjunction with a set of catalog template files.

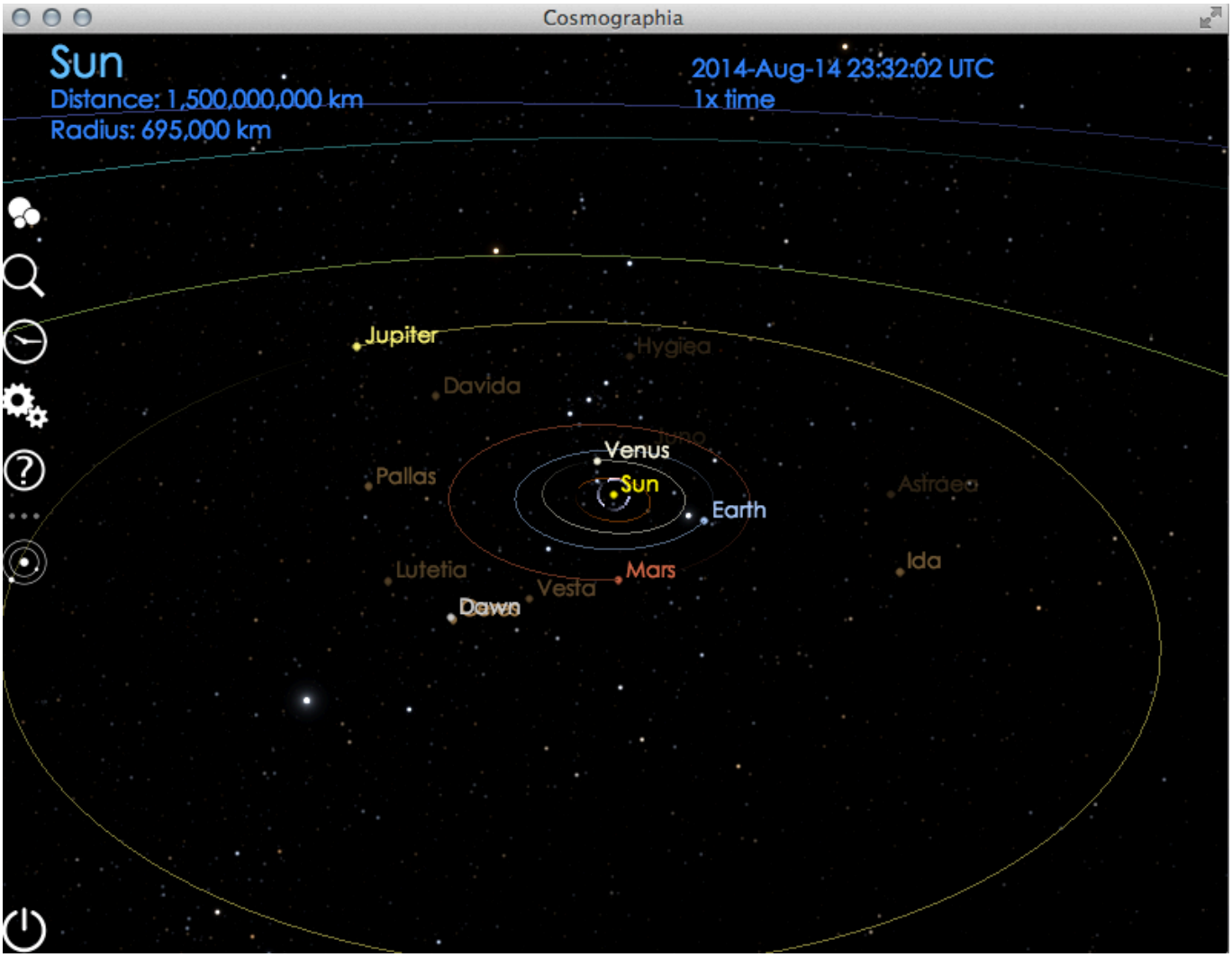
The following links provide more information about Cosmographia, SPICE, and JSON:

- SPICE-enhanced Cosmographia download packages may be found at the NAIF website http://naif.jpl.nasa.gov/naif/cosmographia.html
- User's Guide for SPICE-enhanced Cosmographia may be found at http://cosmoguide.org
- Information on SPICE data may be found at the NAIF website http://naif.jpl.nasa.gov/naif/aboutspice.html
- Basic information on JSON may be found at http://www.json.org

A CAUTIONARY NOTE

The Cosmographia tool and the means for using SPICE data in Cosmographia are both works in progress. Using these offerings will require some patience and experimentation. The user will likely find a number of non-intuitive aspects of operation, and some documentation that is incomplete or otherwise inadequate.

The NAIF Group solicits your feedback on all aspects, including suggestions on how Cosmographia with SPICE might become more useful. Whether or not any fixes or improvements can and will be made is not yet clear.



Default Cosmographia Start-up Screen

II. SPICE Catalog file Specifications

Cosmographia uses catalog files to define solar system objects and their properties. Catalog files for many solar system natural bodies come pre-packaged with the Cosmographia application and use ephemeris, orientation, and shape models provided with or built into the program. Cosmographia users can create and load into the program additional catalog files defining new objects or redefining existing objects. These new catalog files can define

- sets of SPICE data to be used,
- spacecraft, with spacecraft trajectory and orientation provided by SPICE,
- sensors flown on the spacecraft ,with sensor field-of-view (FOV) geometry provided by SPICE,
- sensor observation times ,
- new natural bodies with body trajectory and orientation provided by SPICE.

In the catalog descriptions that follow we use the Cassini mission as an example. Each description discusses just a portion of the complete catalog file. In an appendix at the end we provide the complete version of each Cassini catalog file. In the JSON syntax, names are shown in blue and values are shown in red (character strings) and green (numbers).

Because the descriptions in this User's Guide are broken into chunks, the appearance of JSON syntactical delimiters may seem strange (unbalanced). Look at the example files at the end, or the complete catalog templates, to get a clean, complete view of JSON contents and syntactical structure.

Horizontal rules are used to delimit catalog file “chunks” from descriptions. The descriptions explaining a particular “chunk” normally follow that “chunk”.

In the examples shown in this User's Guide and the associated template files, not all available Cosmographia directives are illustrated. Refer to the on-line Cosmographia documentation to see the full suite of available directives. But also be aware that Cosmographia developers at and outside JPL might be adding still other Cosmographia directives as the use of Cosmographia grows.

1. CATALOG FILE SETTING UP USE OF SPICE DATA

A "SPICE data" catalog file specifies all the SPICE kernels that need to be loaded to run visualization for a particular mission. Start constructing this file using the provided "SPICE data" catalog template (spice_TEMPLATE.json). Note that you can load SPICE meta-kernels, as shown in this example, or individual SPICE kernels.

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "spiceKernels": [
      "kernels/cas_1997_v15.tm",
      "kernels/cas_1998_v15.tm",
      "kernels/cas_1999_v15.tm",
      "kernels/cas_2000_v17.tm",
      "kernels/cas_2001_v18.tm",
      "kernels/cas_2002_v17.tm",
      "kernels/cas_2003_v17.tm",
      "kernels/cas_2004_v17.tm",
      "kernels/cas_2005_v18.tm",
      "kernels/cas_2006_v17.tm",
      "kernels/cas_2007_v15.tm",
      "kernels/cas_2008_v13.tm",
      "kernels/cas_2009_v10.tm",
      "kernels/cas_2010_v09.tm",
      "kernels/cas_2011_v08.tm",
      "kernels/cas_2012_v05.tm",
      "kernels/cas_2013_v02.tm"
   ]
}
```

version
The current version of Cosmographia, 1.0 as of July 15, 2014.
***Example**: "version": "1.0"*

name
The internal name of the catalog file. Any desired descriptive string can be assigned to this parameter, but using the name of the mission in that string is recommended to maintain concurrency throughout all files for the same mission.
***Example**: "name": "Cosmographia CASSINI Example"*

spiceKernels
Names of the SPICE kernels and/or meta-kernels to be used in this visualization. The total number of SPICE kernels is limited only by SPICE capabilities. Loading a large number of SPICE kernels

(hundreds or thousands) lengthens loading time but normally does not affect visualization speed. Note the following:

- Use and maintenance of this catalog file tends to be easier with the use of SPICE meta-kernels rather than individual SPICE files.
- Multiple kernel/meta-kernel names in the list must be separated by commas (this follows the JSON convention for an array).
- The kernel/meta-kernel paths can be absolute or relative to the directory in which the "SPICE data" catalog file resides.
- Individual kernels and meta-kernels provided in the list must be loadable from the directory in which this catalog file resides. For meta-kernels this means that kernels included in them must use correct relative and/or absolute paths.
- When using SPICE data from the PDS' NAIF Node archive, multiple meta-kernel versions for the mission, or for each year of the mission, may exist. In this case, use only the latest version for the mission or a given year.

Single File Example*: "spiceKernels": "cas_1997_v15.tm"*
Multi File Example: *"spiceKernels": ["cas_1997_v15.tm", "cas_1998_v15.tm"]*
Relative Path Example: *"spiceKernels": ["kernels/cas_1997_v15.tm", "kernels/cas_1998_v15.tm"]*
Absolute Path Example: *"spiceKernels": ["/CASSINI/kernels/mk/cas_1997_v15.tm", "/CASSINI/kernels/mk/cas_1998_v15.tm"]*

2. CATALOG FILE DEFINING A SPACECRAFT

A "spacecraft" catalog file defines the spacecraft trajectory, orientation, appearance, and trajectory plot parameters for a spacecraft. Start constructing this file using the provided "spacecraft" catalog template (spacecraft_TEMPLATE.json).

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "spacecraft",
         "name": "Cassini",
         "startTime": "1997-10-15 09:26:08.390 UTC",
         "endTime": "2015-08-01 01:58:52.000 UTC",
```

version
The current version of Cosmographia, 1.0 as of July 15, 2014.
***Example**: "version": "1.0"*

name
The internal name of the catalog file. Any desired descriptive string can be assigned to this parameter, but using the name of the mission in that string is recommended to maintain concurrency throughout all files for the same mission.
***Example**: "name": "Cosmographia CASSINI Example"*

items: class
Set to "spacecraft" in spacecraft catalog files.
***Example**: "class": "spacecraft"*

items: name
The name that will be used to identify the spacecraft within Cosmographia (i.e. this is the Cosmographia name of the spacecraft).
***Example**: "name": "Cassini"*

items: startTime
items: endTime
The spacecraft "lifetime" span within Cosmographia. These times are optional. For on-going missions either endTime or both startTime and endTime can be omitted. If these items are present, they can be set to the start and stop times of the period for which spacecraft trajectory and orientation data are available in SPICE kernels listed in the catalog file defining SPICE data. These start and stop boundaries can be determined for example using the SPICE Toolkit's "brief" and

"ckbrief" utility programs. When SPICE data are provided via one or more meta-kernels (*.tm files), run "brief" and "ckbrief" as follows:

```
% brief -a -t -utc *.tm
…
% ckbrief -a -t -utc -n *.tm
…
```

and pick from the summary output the earliest and latest UTC times for which both the spacecraft trajectory and orientation are available.
***Example**: "startTime": "1997-10-15 09:26:08.390 UTC",*
"endTime": "2015-08-01 01:58:52.000 UTC"

```
"center": "Saturn",
"trajectory": {
   "type": "Spice",
   "target": "Cassini",
   "center": "Saturn"
},
```

---

items: center
The Cosmographia name for the center of the spacecraft's trajectory plot. This parameter will often be set to "Sun" or the Cosmographia name of the body that the spacecraft orbits. For a spacecraft that changes its center of motion the **Arcs** construct can be used to specify different centers over different spans of time. An example of using **Arcs** can be seen in the Examples section of this User's Guide.
***Example**: "center": "Sun"*

items: trajectory: type
Set to "Spice" to make Cosmographia use SPICE data to compute the spacecraft trajectory.
***Example**: "type": "Spice"*

items: trajectory: target
The SPICE name of the spacecraft. This name can be determined by looking at the NAIF_IDS Required Reading document (naif_ids.req), the NAIF name/ID mapping definitions section of the project's Frames Kernel (FK) file (*.tf), or the trajectory data summary produced by the "brief" utility program.
***Example**: "target": "Cassini"*

items: trajectory: center

The SPICE name for the center of the spacecraft's trajectory plot. This must be analogous to the Cosmographia name for the center defined by the *items: center* parameter and should be set to "Sun" or the SPICE name of the body that the spacecraft orbits.
***Example**: "center": "Sun"*

```
"bodyFrame": {
   "type": "Spice",
   "name": "CASSINI_SC_COORD"
},
```

---

items: bodyFrame: type
Set to "Spice" to make Cosmographia use SPICE data to compute spacecraft orientation.
***Example**: "type": "Spice"*

items: bodyFrame: name
The SPICE name of the spacecraft reference frame. This name can be determined by looking at the project's Frames Kernel (FK) file or the orientation data summary produced by the "ckbrief" utility program run with the "-n" option.
***Example**: "fromFrame": "CASSINI_SC_COORD"*

```
"geometry": {
   "type": "Mesh",
   "meshRotation": [
      0,
      0,
      -0.70710677,
      0.70710677
   ],
   "size": 0.005,
   "source": "models/cassini/Cassini_no_Huygens_03.3ds"
},
```

---

items: geometry: type
Set to "Mesh" when a spacecraft 3D model is available in one of the formats supported by Cosmographia (".3ds" or ".cmod").
***Example**: "type": "Mesh"*

items: geometry: meshRotation

The quaternion rotating the spacecraft 3D model to correctly align it with the axes of the spacecraft reference frame specified by the *items: bodyFrame: name* parameter. This quaternion is specified in SPICE format (s,v1,v2,v3).
***Example**: "meshRotation": [0, 0, -0.70710677, 0.70710677]*

items: geometry: size
The size of the displayed model. Cosmographia will scale the 3D model to fit inside a sphere with this radius, in kilometers. If this item is omitted, the model will not be scaled and will be shown with the dimensions specified by the 3D model file.
***Example**: "size": 0.005*

items: geometry: source
The path to the spacecraft 3D model file. This path can be absolute or relative to the directory in which the spacecraft catalog file resides. ".cmod" and ".3ds" are the 3D model file formats supported by Cosmographia.
***Example**: "source": "models/cassini/Cassini_no_Huygens_03.3ds"*

```
            "label": {
               "color": [
                  0.6,
                  1,
                  1
               ],
            "fadeSize": 1000000,
            "showText": true
            },
            "trajectoryPlot": {
               "color": [
                  0.6,
                  1,
                  1
               ],
               "lineWidth": 1,
               "duration": "14 d",
               "lead": "3 d",
               "fade": 1,
               "sampleCount": 100
            }
         }
      ]
}
```

---

items: label: color

The color of the spacecraft name label shown in Cosmographia. The color is based on the RGB color scheme with each component defined on the 0-1 scale. Note that Hex color codes can be used instead of RGB integers, for example "#ff0000" for red.
***Example**: "color": [0.6, 1, 1]*

items: label: fadeSize
The distance (in km) of the object relative to the observer at which the label text will fade from opaque to transparent. If omitted, Cosmographia will try to set the fadeSize to the size of the object's trajectory relative to its central body over an orbit. Since many objects do not have well defined orbits, the program's guess may not be what you expect.
***Example**: "fadeSize": 1000000*

items: label: showText
The logical flag that determines whether the label text should be displayed in Cosmographia. If omitted, this value defaults to true.
***Example**: "showText": true*

items: trajectoryPlot: color
The color of the spacecraft trajectory line shown in Cosmographia. The color is based on the RGB color scheme with each component defined on the 0-1 scale. Note that Hex color codes can be used instead of RGB integers, for example "#ff0000" for red.
***Example**: "color": [0.6, 1, 1]*

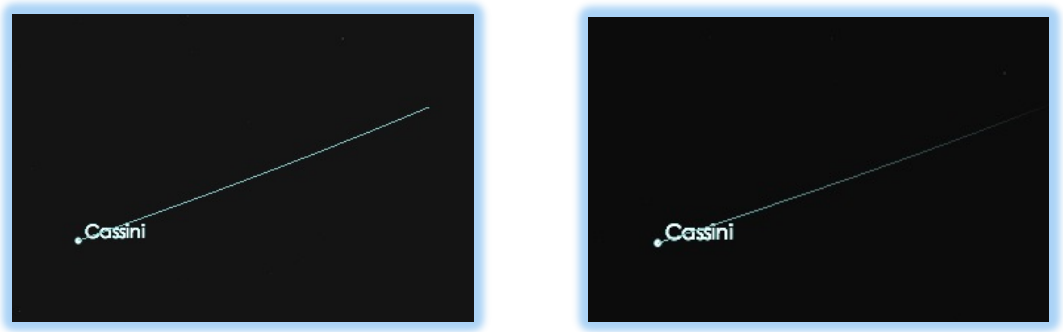
items: trajectoryPlot: lineWidth
The line width of the trajectory plot in pixels. If omitted this value is set to 1.0.
***Example**: "lineWidth": 1*

items: trajectoryPlot: duration
The time duration for which the spacecraft trajectory line will be visible. To get a meaningful trajectory line this parameter should be set to approximately one orbital period, which may be more than a year for a spacecraft in Sun-centric cruise or just a few hours for a spacecraft orbiting a planet. Durations can be specified in a variety of units – years ("y" or "a"), days ("d"), hours ("h"), minutes ("m"), seconds ("s"), and even milliseconds ("ms") – designated by a single or a double character token following a number.
***Example**: "duration": "14 d"*

items: trajectoryPlot: lead
The amount of time in advance of the current time in Cosmographia that the trajectory of the object should be plotted Durations can be specified in a variety of units – years ("y" or "a"), days ("d"), hours ("h"), minutes ("m"), seconds ("s"), and even milliseconds ("ms") – designated by a single or a double character token following a number.
***Example**: "lead": "3 d"*

items: trajectoryPlot: fade

The degree of the fading effect on the trajectory plot visualization, based on a 0-1 scale, with 0 being completely opaque to 1 fading the most while still being visible.
***Example**: "fade": 1*



This is an example of *items: trajectoryPlot: fade* set at 0.0 (left) and 1.0 (right)

items: trajectoryPlot: sampleCount

This is an advanced feature of the trajectory plot that allows the user to define the number of points to interpolate between when drawing the trajectory. The interpolation points, known as samples, are evenly spaced across the plot duration. If omitted, Cosmographia chooses 100 samples. Cosmographia restricts the range of sampleCount to be between 100 and 200000.
***Example**: "sampleCount": 100*

3. CATALOG FILE DEFINING A SENSOR

A "sensor" catalog file defines an instrument's location, orientation, and appearance. It is recommended to create a separate sensor catalog file for each sensor with a single field-of-view (FOV), and for each individual FOV for a sensor with multiple FOVs. The example used here is for the Cassini ISS Narrow Angle Camera, which has a single FOV.

A non-intuitive aspect of a sensor definition is that the target body the sensor will be pointing at is a part of the definition. Additional sensor catalog files – one for each sensor-target combination – must be created if you wish Cosmographia to be able to display a given sensor's observations of additional targets.

Start constructing this file using the provided "sensor" catalog template (sensor_TEMPLATE.json).

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "sensor",
         "name": "CAS_ISS_NAC",
         "parent": "Cassini",
```

version
The current version of Cosmographia, 1.0 as of July 15, 2014.
***Example**: "version": "1.0"*

name
The internal name of the catalog file. Any desired descriptive string can be assigned to this parameter, but using the name of the mission in that string is recommended to maintain concurrency throughout all files for the same mission.
***Example**: "name": "Cosmographia CASSINI Example"*

items: class
Set to "sensor" in sensor catalog files.
***Example**: "class": "sensor"*

items: name
The name that will be used to identify the sensor within Cosmographia (or the Cosmographia name of the sensor).
***Example**: "name": "CAS_ISS_NAC"*

items: parent
The Cosmographia name of the spacecraft to which the sensor is attached. This name must be identical to the name defined by the *items: name* parameter of the corresponding spacecraft catalog file.
***Example**: "parent" : "Cassini"*

```
"startTime": "1997-10-15 09:26:08.390 UTC",
"endTime": "2015-08-01 01:58:52.000 UTC",
"center": "Cassini",
"trajectoryFrame": {
   "type": "BodyFixed",
   "body": "Cassini"
},
```

---

items: startTime
items: endTime
The sensor "lifetime" span within Cosmographia. These times are optional. For on-going missions either endTime or both startTime and endTime can be omitted. If these items are present, the time span is normally set to be the same as the spacecraft "lifetime" span defined in the corresponding spacecraft catalog file. Note though that this parameter does not necessarily mean that the sensor's FOV frustum will be shown during this whole "lifetime" period. If the *items: geometry: onlyVisibleDuringObs* parameter described below is set to true, the FOV frustum will only be shown during observation time intervals defined for this sensor-target combination in an observation catalog file.
***Example**: "startTime": "1997-10-15 09:26:08.390 UTC",*
"endTime": "2015-08-01 01:58:52.000 UTC"

items: center
The Cosmographia name of the spacecraft to which the sensor is attached. This should be identical to *items: parent.*
***Example**: "center": "Cassini"*

items: trajectoryFrame: type
The type of reference frame the sensor's trajectory is defined in. For a sensor attached to a spacecraft this parameter should be set to "BodyFixed".
***Example**: "type": "BodyFixed"*

items: trajectoryFrame: body
The Cosmographia name of the spacecraft to which the sensor is attached. This should be identical to *items: parent.*
***Example**: "body": "Cassini"*

```
"geometry": {
   "type": "Spice",
   "instrName": "CASSINI_ISS_NAC",
   "target": "Saturn",
   "range": 45000,
   "rangeTracking": true,
   "frustumColor": [
      0,
      1,
      1
   ],
   "frustumOpacity": 0.3,
```

items: geometry: type
Set this to "Spice" to make Cosmographia use SPICE data to define the sensor FOV shape and size.
***Example**: "type": "Spice"*

items: geometry: instrName
The SPICE name of the sensor. This name can be determined by examining the sensor's IK file and/or the NAIF name/ID mapping definitions section of the project's FK file and finding the name that corresponds to the NAIF ID code that is used in the IK keywords defining the sensor FOV.
***Example**: "instrName": "CASSINI_ISS_NAC"*

items: geometry: target
The Cosmographia name of the body that the sensor will be targeting. Each sensor-target pair is treated by Cosmographia as one object. For this reason it is recommended to create a separate catalog file for each sensor-target pair. For instance, two separate but nearly identical sensor catalog files would be needed for ISS-Saturn and ISS-Titan sensor-target pairs, with the first file setting target to "Saturn" and the second setting it to "Titan".
***Example**: "target": "Saturn"*

items: geometry: range
The length of the sensor's FOV frustum shown by Cosmographia, in kilometers. This item is ignored if the rangeTracking option is on.
Example: "range": 45000

items: geometry: rangeTracking
The logical flag turning on (true) or off (false) target range tracking.

- Turning on the rangeTracking option directs Cosmographia to dynamically set the frustum length equal to the distance between the spacecraft and the target body.

- If this flag is true, the *items: geometry: range* parameter is not necessary and if present, is ignored by the program.
- Setting this parameter to true is recommended for visual appeal.
- Setting this parameter to false and instead defining the frustum length by the *items: geometry: range* parameter is recommended for cases when displaying a frustum with a fixed length is desired.

***Example**: "rangeTracking": true*

items: geometry: frustumColor

The color of the sensor's FOV frustum shown in Cosmographia. The color is based on the RGB color scheme with each component defined on the 0-1 scale. Note that Hex color codes can be used instead of RGB integers, for example "#ff0000" for red.

***Example**: "frustumColor" : [0, 1, 1]*

items: geometry: frustumOpacity

The visibility of the sensor's FOV frustum on the 0 to 1 scale, with 0 being completely transparent to 1 being completely opaque.

***Example**: "frustumOpacity": 0.3*

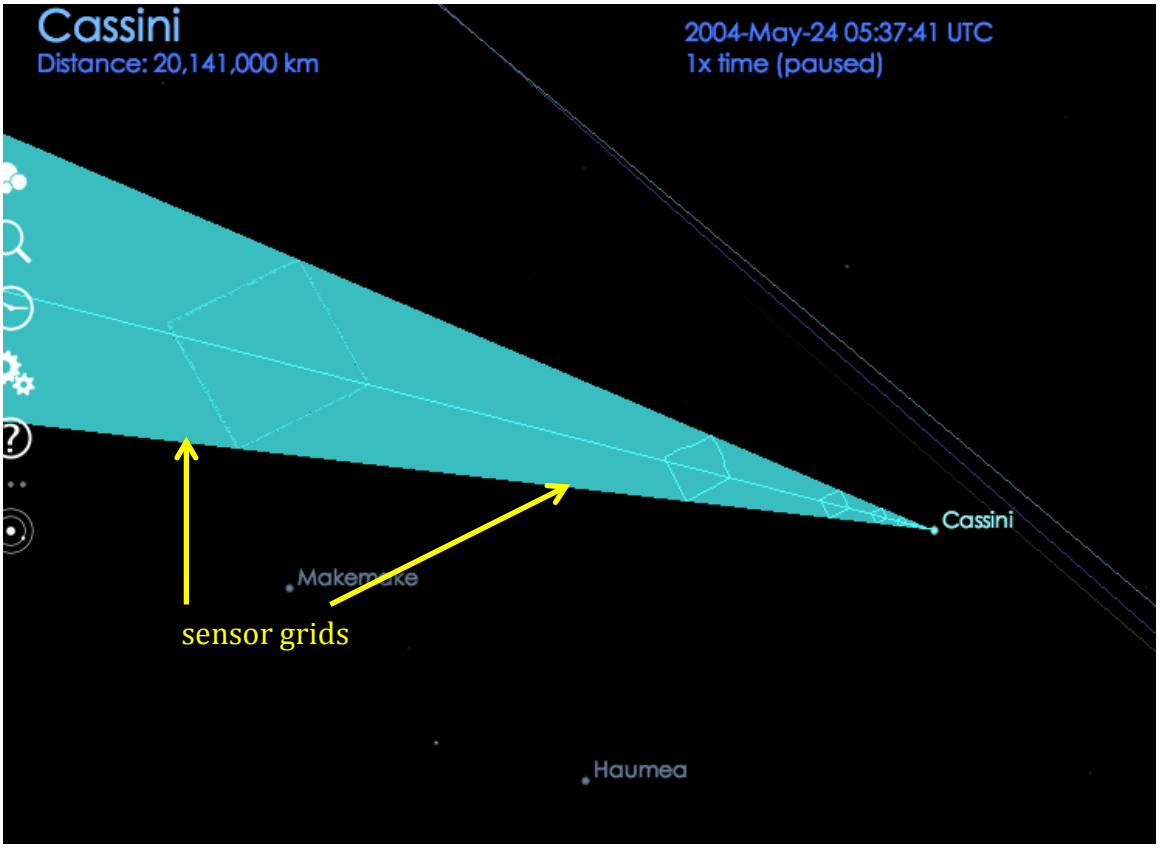
```
            "gridOpacity": 1,
            "footprintOpacity": 0.8,
            "sideDivisions": 3000,
            "onlyVisibleDuringObs": false
        }
    }
  ]
}
```

---

items: geometry: gridOpacity

The visibility of the sensor's FOV frustum grids on the 0 to 1 scale, with 0 being transparent to 1 being completely opaque.

***Example**: "gridOpacity": 1,*



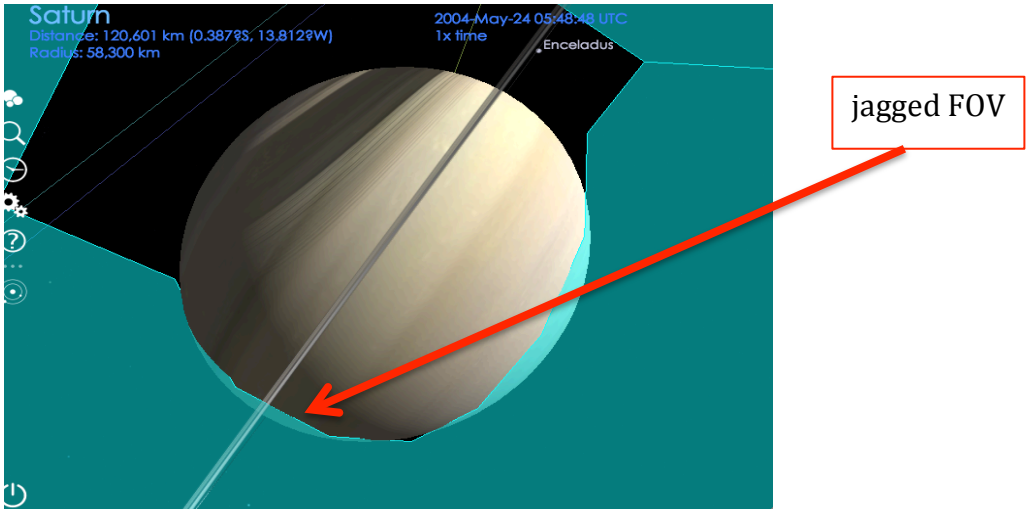
Example: Cassini's ISS NAC sensor with gridOpacity set to 1

items: geometry: footprintOpacity
The visibility of the sensor's FOV footprint on a 0 to 1 scale, with 0 being transparent to 1 being completely opaque. The sensor's FOV footprint is the outline on the far end of the sensor's frustum.
***Example**: "footprintOpacity": 0.8,*

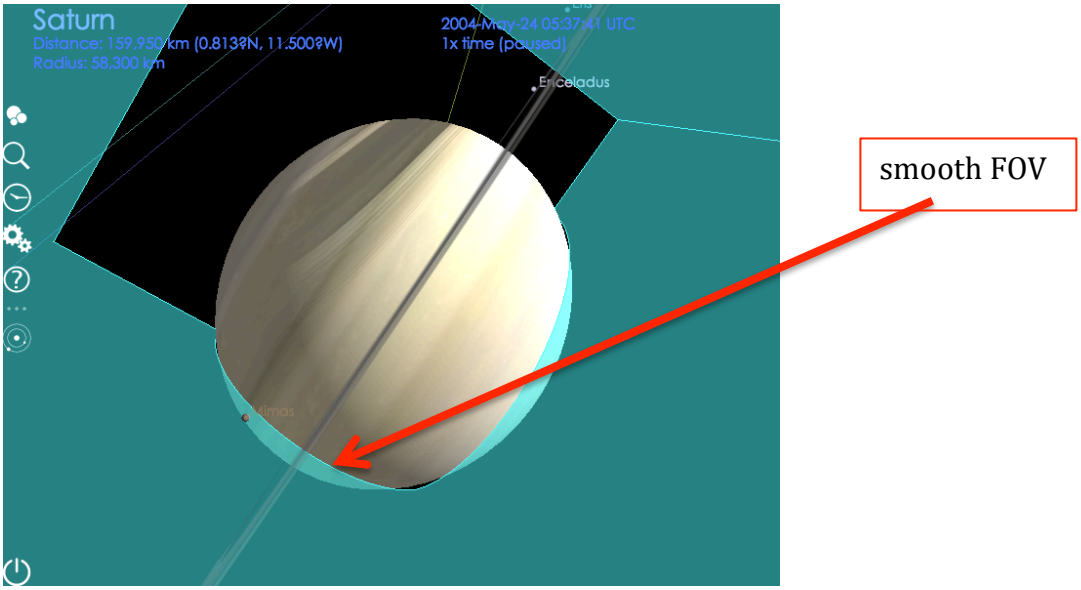
items: geometry: sideDivisions
The number of points plotted per each side of the sensor's FOV frustum.

- A larger number of points will result in a smoother sensor FOV side rendering but will be more likely to cause slow Cosmographia rendering.
- In general, 125 is a good default number for optimal visuals and running.
- If the sideDivisions parameter is not defined in the catalog file, Cosmographia will use 125 by default.
- Reducing the number of side divisions can help speed up Cosmographia in the case of lagging.

***Example**: "sideDivisions": 3000*



This is a visualization of Cassini's ISS NAC using 5 side divisions



This is a visualization of Cassini's ISS NAC using 3000 side divisions

items: geometry: onlyVisibleDuringObs

The logical flag restricting the sensor's FOV frustum rendering to only times when the sensor is taking observations. If true, the sensor's FOV frustum, grids, and footprints will only be visible during the times the sensor is taking observations as specified by an observations catalog file. If false, the sensor's FOV frustum, grids, and footprints will be visible for the whole sensor "lifetime" span specified in the *items: startTime* and *items: endTime* specified earlier in the sensor file.
Example*: "onlyVisibleDuringObs": false*

4. CATALOG FILE FOR DEFINING AN OBSERVATION

An "observations" catalog file defines the observation times and footprint appearance for a series of observations of a target taken by a sensor. For observations defined by an observation catalog file to be recognized and rendered by Cosmographia a spacecraft catalog file and a sensor catalog file defining the sensor-target pair of interest must be created and loaded into Cosmographia before loading the observation catalog file.

Start constructing this file using the provided "observation" catalog template (obs_TEMPLATE_groups.json).

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "observation",
         "name": "CASSINI_ISS_NAC_OBSERVATION",
         "startTime": "1997-10-15 09:26:08.390 UTC",
         "endTime": "2015-08-01 01:58:52.000 UTC",
```

version
The current version of Cosmographia, 1.0 as of July 15, 2014.
***Example**: "version": "1.0"*

name
The internal name of the catalog file. Any desired descriptive string can be assigned to this parameter, but using the name of the mission in that string is recommended to maintain concurrency throughout all files for the same mission.
***Example**: "name": "Cosmographia CASSINI Example"*

items: class
Set to "observation" in observation catalog files.
***Example**: "class": "observation"*

items: name
The name that will be used to identify this observation within Cosmographia (or the Cosmographia name of the observation).
***Example**: "name": "CASSINI ISS NAC OBSERVATION"*

items: startTime
items: endTime
The observation's "lifetime" span within Cosmographia. These times are optional. For on-going missions either endTime or both startTime and endTime can be omitted. If these items are present, this time span is normally set to be the same as the sensor "lifetime" span defined in the corresponding sensor catalog file but it can also be made shorter if desired. Note, though, that this time span does not determine the actual times for which the observation footprints or swaths are computed. Instead the actual times are specified in the *items: geometry: groups: startTime* and *endTime* parameters described below and the observation "lifetime" span defined here must be equal to or be wider than the time span of these actual observations.
***Example**: "startTime": "1997-10-15 09:26:08.390 UTC",*
"endTime": "2015-08-01 01:58:52.000 UTC"

```
"center": "Saturn",
"trajectoryFrame": {
   "type": "BodyFixed",
   "body": "Saturn"
},
"bodyFrame": {
   "type": "BodyFixed",
   "body": "Saturn"
},
```

---

items: center
The Cosmographia name of the target body. This should be identical to the *items: geometry: target* value specified in the corresponding sensor catalog file.
***Example**: "center": "Saturn"*

items: trajectoryFrame: type
The type of reference frame the observation footprint is defined in. For observation footprints to appear fixed on the target body surface this parameter should be set to "BodyFixed".
***Example**: "type": "BodyFixed"*

items: trajectoryFrame: body
The Cosmographia name of the target body. This should be identical to the *items: geometry: target* value specified in the corresponding sensor catalog file.
***Example**: "body": "Saturn"*

items: bodyFrame: type
The type of reference frame the observation footprint is defined in. For observation footprints to appear fixed on the target body surface this parameter should be set to "BodyFixed".

***Example**: "type": "BodyFixed"*

items: bodyFrame: body
The Cosmographia name of the target body. This should be identical to the *items: geometry: target* value specified in the corresponding sensor catalog file.
***Example**: "body": "Saturn"*

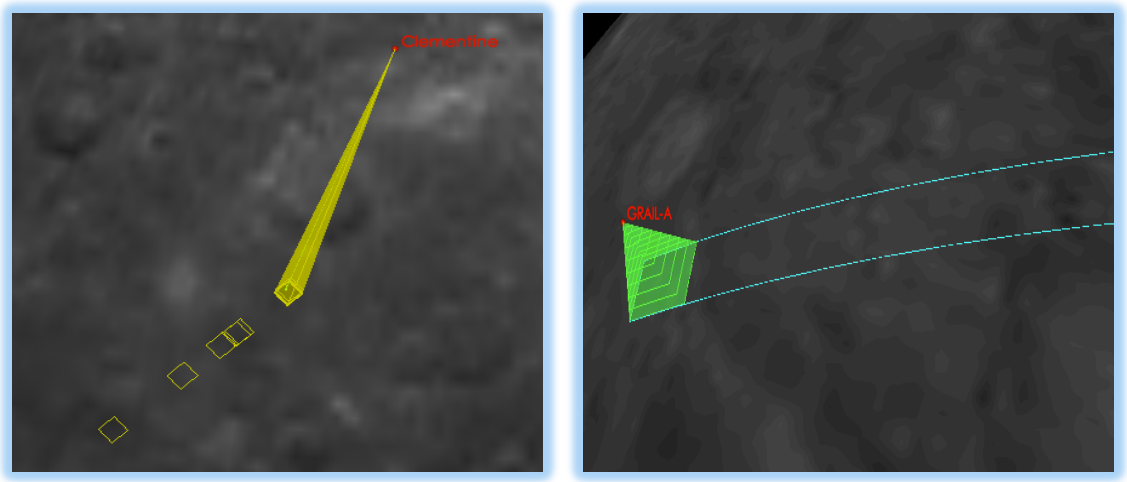
```
"geometry": {
   "type": "Observations",
   "sensor": "CAS_ISS_NAC",
   "groups": [
      {
         "startTime": "2004-05-24 05:48:03.043 UTC",
         "endTime": "2004-05-24 05:48:08.643 UTC",
         "obsRate": 0
      },
      {
         "startTime": "2004-05-24 05:53:03.069 UTC",
         "endTime": "2004-05-24 05:53:05.669 UTC",
         "obsRate": 0
      }
   ],
```

---

items: geometry: type
Set this to “Observations” to render observation swaths or footprints.
***Example**: "type": "Observations"*

items: geometry: sensor
The Cosmographia name of the sensor taking the observations defined in this file.
This value should be the same as the *items: name* value in the corresponding sensor file.
***Example**: "sensor": "CAS_ISS_NAC"*

items: geometry: groups: startTime
items: geometry: groups: endTime
items: geometry: groups: obsRate
The start and stop time and the observation rate for each actual observation in the series. The observation rate determines the rate, in seconds, at which footprints will be drawn between the start and stop times. Setting the observation rate to zero will create a continuous swath rather than a series of footprints. Multiple actual observation time/rate triplets with distinct start/stop boundaries and same or different rates can be specified within a group block.
***Example**: "startTime" :"2004-05-24 05:48:03.043 UTC",*
"endTime" :"2004-05-24 05:48:08.643 UTC"
“obsRate” : 0



This is an Example of observation footprints (left) and swaths (right)

```
            "footprintColor": [
               1,
               0.5,
               0
            ],
            "footprintOpacity": 0.4,
            "showResWithColor": false,
            "alongTrackDivisions": 500,
            "sideDivisions": 125,
            "shadowVolumeScaleFactor": 1.75,
            "fillInObservations": false
         }
      }
   ]
}
```

---

items: geometry: footprintColor
The color of the footprints shown in Cosmographia. The color is based on the RGB color scheme with each component defined on the 0-1 scale. Note that Hex color codes can be used instead of RGB integers, for example "#ff0000" for red.
***Example**: "footprintColor": [1.0, 0.5, 0.0]*

items: geometry: footprintOpacity
The visibility of the footprints on a 0 to 1 scale, with 0 being transparent to 1 being completely opaque.
***Example**: "footprintOpacity": 0.4*

items: geometry: showResWithColor
The logical flag enabling the footprints to dynamically change color depending on the distance between the target body and the spacecraft. Customization of distance-color correspondence can be done using the *items: geometry: colorScheme* parameter not described in this User's Guide.
Example*: "showResWithColor": false*

items: geometry: alongTrackDivisions
The number of segments that the side of a swath will be divided into during visualization. Decreasing this number from its default of 1000 can improve rendering speed.
Example*: "alongTrackDivisions": 500*

items: geometry: sideDivisions
The number of points plotted per each side of a footprint. A larger number of points will result in a smoother sensor FOV side rendering but will be more likely to cause slow Cosmographia rendering. In general, 125 is a good default number for optimal visuals and running. If the *sideDivisions* parameter is not defined in the catalog file, Cosmographia will use 125 by default. Reducing the number of side divisions can help speed up Cosmographia in the case of lagging..
Example*: "sideDivisions": 125*

items: geometry: shadowVolumeScaleFactor
The factor scaling the length of the shadow volume used in rendering filled-in observations. Using a larger factor is helpful in ensuring proper footprint rendering on oblong bodies.
Example*: "shadowVolumeScaleFactor": 1.75*

items: geometry: fillInObservations
The logical flag enabling filling of the footprints with color. If true, observations will appear filled in with color. If false, observations will be outlined.
Example*: "fillInObservations": false*

5. CATALOG FILE DEFINING A NATURAL BODY

A "natural body" catalog file defines the natural body trajectory, orientation, appearance, and trajectory plot parameters for a natural body of interest. Natural body catalog files can be created to define new bodies. It is possible but usually not necessary to redefine existing bodies because majority of Solar system bodied in the SPICE-enhanced version of Cosmographia already use SPICE trajectory and orientation data. The catalog file presented in this section defines parameters for Saturn just to illustrate how its is done.

Start constructing this file using the provided "body" catalog template (body_TEMPLATE_globe.json).

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "planet",
         "name": "Saturn",
```

version
The current version of Cosmographia, 1.0 as of July 15, 2014.
***Example**: "version": "1.0"*

name
The internal name of the catalog file. Any desired descriptive string can be assigned to this parameter, but using the name of the mission in that string is recommended to maintain concurrency throughout all files for the same mission.
***Example**: "name": "Cosmographia CASSINI Example"*

items: class
Set this to one of the following applicable class values: "planet," "satellite", "asteroid", "dwarf planet", "reference point", "other."
***Example**: "class": "planet"*

items: name
The name that will be used to identify this natural body within Cosmographia (or the Cosmographia name of the body). When re-defining an existing body, this name must exactly match the name already used by Cosmographia for the body.
***Example**: "name : "Saturn"*

```
        "mass": "95.152 Mearth",
        "density": 0.687,
```

items: mass
The mass of the body in kg (use "kg" as units), or in relation to the mass of the Earth (use "Mearth" as suffix).
***Example**: "mass": "95.152 Mearth"*

items: density
The density of the object in g/cm^3.
***Example**: "density": 0.687*

```
        "center": "Sun",
        "trajectory" : {
           "type": "Spice",
           "target": "SATURN",
           "center": "SUN"
        },
```

items: center
The Cosmographia name for the center of the body's trajectory plot. This parameter will usually be set to "Sun" or to the Cosmographia name of the planet or planet barycenter that the body orbits.
***Example**: "center": "Sun"*

items: trajectory: type
Set this to "Spice" to make Cosmographia use SPICE data to compute body's trajectory.
***Example**: "type": "Spice"*

items: trajectory: target
The SPICE name of the natural body. This name can be determined by looking at the NAIF_IDS Required Reading document (naif_ids.req), the NAIF name/ID mapping definitions section of the project's Frames Kernel (FK) file (*.tf), or the trajectory data summary produced by the "brief" utility program.
***Example:** "target": Saturn"*

items: trajectory: center
The SPICE name for the center of the body's trajectory plot. This must be analogous to the Cosmographia name for the center defined by the *items: center* parameter and should be set to "Sun" or the SPICE name of the planet or planet barycenter that the body orbits.
***Example**: "center": "Sun"*

```
"bodyFrame": {
   "type": "Spice",
   "name": "IAU_SATURN"
},
```

items: bodyFrame: type
Set this to "Spice" to make Cosmographia use SPICE data to compute the body's orientation.
Example*: "type": "Spice"*

items: bodyFrame: name
The SPICE name of the body's body-fixed reference frame. This name can be determined by looking at the Frames Required Reading document or the project's FK file.
Example*: "fromFrame": "IAU_SATURN"*

There are multiple ways to define the "geometry" of a natural body. The three most likely to be of use are "Globe", "Mesh" and "Rings". Choose one to use as your geometry block.

- Globe: uses a tri-axial ellipsoid to model the body
- Mesh: uses an imported 3D shape to model the body and is typically used for small, irregularly shaped bodies
- Rings: uses an inner and outer radius to model a ring or ring set

Here we have an example using "Globe."

```
"geometry": {
   "type": "Globe",
   "radii": [
      60268,
      60268,
      54364
   ],
   "baseMap": "textures/saturn.jpg"
},
```

items: geometry: type
The object shape type. Using "Globe" as the type is recommended for spherical or ellipsoidal astronomical objects.
Example: *"type": "Globe"*

items: geometry: radii

The radii of the tri-axial ellipsoid used as the body's shape. The radii are measured in kilometers from the body's center of geometry along x, y, and z axes of the body reference frame. The radii can be found in a Planetary Constants kernel (PCK).
Example: *"radii": [60268, 60268 , 54364]*

items: geometry: baseMap
The path to the jpg texture file used to overlay the base globe. This path can be absolute or relative to the directory in which the natural body catalog file resides.
Example: *"baseMap": "textures/Saturn.jpg"*

```
            "label": {
               "color": [
                  0.8,
                  1.0,
                  0.5
               ]
            },
            "trajectoryPlot": {
                "duration": "10759.22 d",
                "fade": 0.3
            }
        }
    ]
}
```

---

items: label: color
The color of the body name label displayed in Cosmographia. The color is based on the RGB color scheme with each component defined on a 0-1 scale. Note that Hex color codes can be used instead of RGB integers, for example "#ff0000" for red.
Example*: "color": [0.8, 1, 0.5]*

items: trajectoryPlot: duration
The time duration, in years, for which the spacecraft trajectory line will be visible. To get a meaningful trajectory line this parameter should be set to approximately one orbital period. Durations can be specified in a variety of units – years ("y" or "a"), days ("d"), hours ("h"), minutes ("m"), seconds ("s"), and even milliseconds ("ms") – designated by a single or a double character token following a number.
Example*: "duration": "10759.22 d"*

items: trajectoryPlot: fade
The degree of the fading effect on the trajectory plot visualization, based on a 0-1 scale, with 0 being completely opaque to 1 fading the most while still being visible.
Example*: "fade" : 0.3*

6. CATALOG FILE TO LOAD MULTIPLE FILES

Frequently it is convenient to load multiple catalog files for a given mission together. This can be done creating a "Load" catalog file.

Start constructing this file using the provided "load" catalog template (load_TEMPLATE.json).

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "require": [
      "spice_CASSINI.json",
      "spacecraft_CASSINI.json",
      "sensors/sensor_CASSINI_ISS_NAC-SATURN.json",
      "observations/obs_CASSINI_ISS_NAC-SATURN-0405240548.json"
   ]
}
```

version
The current version of Cosmographia, 1.0 as of July 15, 2014.
***Example**: "version": "1.0"*

name
The internal name of the catalog file. Any desired descriptive string can be assigned to this parameter, but using the name of the mission in that string is recommended to maintain concurrency throughout all files for the same mission.
***Example**: "name": "Cosmographia CASSINI Example"*

require
The list of all catalog files that will be loaded together when this catalog is loaded. Individual catalog files must be listed in this catalog file in proper order. Look at "Loading Order of Files into Cosmographia" in Section 7 of this User's Guide for more information. The paths to catalog files can be absolute or relative to the directory in which the "load" catalog file resides.

***Example**:* *"spice_CASSINI.json",*
"spacecraft_CASSINI.json",
"sensors/sensor_CASSINI_ISS_NAC-SATURN.json",
"observations/obs_CASSINI_ISS_NAC-SATURN-0405240548.json"

7. ARCS FUNCTIONALITY

"Arcs" functionality can be used to make Cosmographia show meaningful trajectory plots when the center of motion for a spacecraft trajectory changes from one body to another, such as from sun-centered (during cruise phase) to planet centered (upon planet close flyby or orbit insertion).

An example of using **Arcs** is found near the end of this User's Guide and in a separate spacecraft **Arcs** catalog template (spacecraft_TEMPLATE_arcs.json).

8. IMPORTANT THINGS TO REMEMBER

Loading Order of Files into Cosmographia
The loading of separate catalog files must be done in order of information dependency. For instance, the SPICE data catalog file for a mission must be loaded prior to any spacecraft, sensor, or observation files since each of the latter require Cosmographia to read SPICE data.

In general, files should be loaded in the following order:

1. SPICE data catalog file (names the SPICE kernels to be used)
2. Spacecraft catalog file(s)
3. Natural body catalog files(s) (if defining new natural bodies to be visualized by Cosmographia)
4. Sensor catalog file(s)
5. Observation catalog file(s)

An optional "load" catalog file may be used to load all needed catalog files. If used, the ordering of lines in the "load" catalog file must follow the loading order rules mentioned above. An example of a "load" catalog file is provided earlier in this User's Guide.

Using Multiple Files of a Given Type
In theory all of the specifications needed to run Cosmographia with SPICE could be provided in a single catalog file, but this would be difficult to construct and to maintain. Using the functional separation shown above – SPICE data, Spacecraft, natural body, Sensor and Observation catalog files – is a better way to organize your inputs. Further, supplying separate files for each sensor-target pair and each observation set is recommended.

Case-Sensitivity
Parameters are case sensitive such as in defining "items: class" and incorrect case usage may cause files to not be loaded properly.

File Placement
Individual SPICE files or meta-kernels must be in the same folder as the corresponding SPICE data catalog file, otherwise a path to the file should be given as an absolute path or a relative to the SPICE data catalog file**.**

The Parameters in Each of the Following Groups Must be *Identical*
This is because these parameters are used to link the objects defined by catalog files to one another. For instance, the spacecraft's "name" must be identical to a sensor's "parent" for Cosmographia to associate the two objects together.

<u>Group 1</u>

- "items" : "name" (in Spacecraft catalog)
- "items" : "parent" (in Sensor catalog)
- "items" : "center" (in Sensor catalog)

- “items” : “trajectoryFrame” : “body” (in Sensor catalog)
- “items” : “bodyFrame” : “body” (in Sensor catalog)

Group 2
- “items” : “name” (Sensor catalog)
- “items” : “geometry” : “sensor” (in Observation catalog)

Group 3
- **“**items” : “geometry” : “target” (in Sensor catalog)
- **“**items” : “center” (in Observation catalog)
- “items” : “trajectoryFrame” : “body” (in Observation catalog)
- “items” : “bodyFrame” : “body” (in Observation catalog)

The Parameters in Each of the Following Groups Must be *Different Versions of Each Other*
This is because these parameters are used to match SPICE bodies to analogous bodies within Cosmographia.

All Parameters are in both Spacecraft and Natural Body Catalog Files

Group 1
- “items” : “center” (Cosmographia name)
- “items” : “trajectory” : “center” (SPICE name)

9. COMMON PROBLEMS AND POSSIBLE SOLUTIONS

If Observations are not showing…
The name assigned for the sensor's target is not correct or the range of the sensor may be too short to reach the sensor's target. To fix the latter, set the *rangeTracking* parameter to 'true' in the sensor file. Observations may also not be appearing if *footprintOpacity* is set to 0.

If Cosmographia is rendering too slowly…
It may help if you:

- decrease the number of *sideDivisions* for sensor and/or observation objects
- decrease the number of observation time intervals (groups)
- decrease the number of *alongTrackDivisions* for swath observations (when obsRate=0)
- decrease *obsRate*
- set *footprintOpacity* or *gridOpacity* to zero in the sensor object

If Cosmographia is crashing after loading a file…
Check to make sure that *sideDivisions* in the sensor file is *not* set to 0 or 1.
If all files seem to be in good shape, try restarting Cosmographia.
NOTE: *Cosmographia will not crash due to any syntax error in catalog files.*

If a "Cannot be Found" error message constantly shows up when loading a file…
This is probably because of a mismatch between any of the following:

- *items: name* in the spacecraft file and *items: parent* in a sensor file
- *items: name* in a sensor file and *geometry: sensor* in an observation file

If sensors are too thin for observations to be seen…
This may happen for sensors such as laser altimeters. In this case, setting the *obsRate* to 0 and assigning a well contrasting color to the footprint color may help provide a clearer visualization of observations.

If sensors are constantly pointing towards empty space…
This may actually be what was happening during the specified time, or this may be caused by lack of SPICE orientation data for the sensor frame.

No Spice Data…
There may be certain time intervals during Cosmographia simulation in which insufficient SPICE data is available to render visuals properly. In this case, a "No Spice Data" warning with the full or abbreviated list of SPICE names of bodies and/or frames will appear across the top of the Cosmographia window and the object associated with this gap may jump to the Sun (as default) or have some other strange orientation or trajectory.

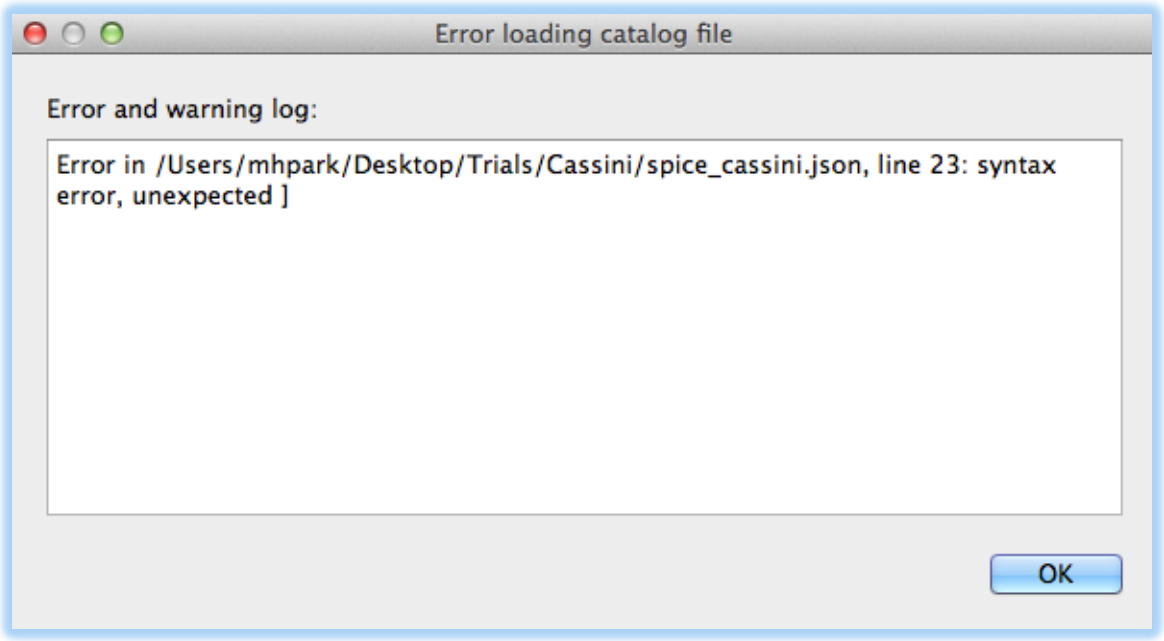
The reason for this gap can be examining SPICE error messages available in "Spice Log" under "File" in Cosmographia.

Common reasons include:

- missing CK file for articulating instruments
- gaps in CK files for spacecraft
- wrong bodyFrame *name*
- wrong *instrName*

Common Syntax Errors ...

Cosmographia will display an error message such as the following if syntax error is disabling proper file loading.



Common syntax errors are:

- misplacement of commas, particularly at the end of a list
- extra brackets, parentheses, or curly brackets
- lack of quotation mark(s)

Catalog Load Warning ...

Cosmographia will display descriptive warning messages during catalog loading when catalogs re-define already existing bodies and use inconsistent SPICE frames in two-step body orientation definitions, and in a few other cases. These warnings don't prevent catalogs from being loaded and Cosmographia from running.

III. COMPLETE EXAMPLES OF CATALOG FILES

Example 1: Files for Cassini Orbiter

Example 1A: Cassini **SPICE** Data Catalog File

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "spiceKernels": [
      "kernels/cas_1997_v15.tm",
      "kernels/cas_1998_v15.tm",
      "kernels/cas_1999_v15.tm",
      "kernels/cas_2000_v17.tm",
      "kernels/cas_2001_v18.tm",
      "kernels/cas_2002_v17.tm",
      "kernels/cas_2003_v17.tm",
      "kernels/cas_2004_v17.tm",
      "kernels/cas_2005_v18.tm",
      "kernels/cas_2006_v17.tm",
      "kernels/cas_2007_v15.tm",
      "kernels/cas_2008_v13.tm",
      "kernels/cas_2009_v10.tm",
      "kernels/cas_2010_v09.tm",
      "kernels/cas_2011_v08.tm",
      "kernels/cas_2012_v05.tm",
      "kernels/cas_2013_v02.tm"
   ]
}
```

Example 1B: Cassini **Spacecraft** Catalog File


```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "spacecraft",
         "name": "Cassini",
         "startTime": "1997-10-15 09:26:08.390 UTC",
         "endTime": "2015-08-01 01:58:52.000 UTC",
         "center": "Saturn",
         "trajectory": {
            "type": "Spice",
            "target": "Cassini",
            "center": "Saturn"
         },
         "bodyFrame": {
            "type": "Spice",
            "name": "CASSINI_SC_COORD"
         },
         "geometry": {
            "type": "Mesh",
            "meshRotation": [
               0,
               0,
               -0.70710677,
               0.70710677
            ],
            "size": 0.005,
            "source": "models/cassini/Cassini_no_Huygens_03.3ds"
         },
         "label": {
            "color": [
               0.6,
               1,
               1
            ],
         "fadeSize": 1000000,
         "showText": true
         },
         "trajectoryPlot": {
            "color": [
               0.6,
               1,
               1
            ],
            "lineWidth": 1,
            "duration": "14 d",
            "lead": "3 d",
            "fade": 1,
```

```
            "sampleCount": 100
        }
    }
  ]
}
```

Example 1C: Cassini **Sensor** Catalog File

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "sensor",
         "name": "CAS_ISS_NAC",
         "parent": "Cassini",
         "startTime": "1997-10-15 09:26:08.390 UTC",
         "endTime": "2015-08-01 01:58:52.000 UTC",
         "center": "Cassini",
         "trajectoryFrame": {
            "type": "BodyFixed",
            "body": "Cassini"
         },
         "geometry": {
            "type": "Spice",
            "instrName": "CASSINI_ISS_NAC",
            "target": "Saturn",
            "range": 45000,
            "rangeTracking": true,
            "frustumColor": [
               0,
               1,
               1
            ],
            "frustumOpacity": 0.3,
            "gridOpacity": 1,
            "footprintOpacity": 0.8,
            "sideDivisions": 3000,
            "onlyVisibleDuringObs": false
         }
      }
   ]
}
```

Example 1D: Cassini **Observation** Catalog File

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "observation",
         "name": "CASSINI_ISS_NAC_OBSERVATION",
         "startTime": "1997-10-15 09:26:08.390 UTC",
         "endTime": "2015-08-01 01:58:52.000 UTC",
         "center": "Saturn",
         "trajectoryFrame": {
            "type": "BodyFixed",
            "body": "Saturn"
         },
         "bodyFrame": {
            "type": "BodyFixed",
            "body": "Saturn"
         },
         "geometry": {
            "type": "Observations",
            "sensor": "CAS_ISS_NAC",
            "groups": [
               {
                  "startTime": "2004-05-24 05:48:03.043 UTC",
                  "endTime": "2004-05-24 05:48:08.643 UTC",
                  "obsRate": 0
               },
               {
                  "startTime": "2004-05-24 05:53:03.069 UTC",
                  "endTime": "2004-05-24 05:53:05.669 UTC",
                  "obsRate": 0
               }
            ],
            "footprintColor": [
               1,
               0.5,
               0
            ],
            "footprintOpacity": 0.4,
            "showResWithColor": false,
            "sideDivisions": 125,
            "alongTrackDivisions": 500,
            "shadowVolumeScaleFactor": 1.75,
            "fillInObservations": false
         }
      }
   ]
}
```

Example 1E: Cassini **Load All** Catalog File

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "require": [
      "spice_CASSINI.json",
      "spacecraft_CASSINI.json",
      "sensors/sensor_CASSINI_ISS_NAC-SATURN.json",
      "observations/obs_CASSINI_ISS_NAC-SATURN-0405240548.json"
   ]
}
```

Example 2: Using Arcs

Example 2: Cassini **Spacecraft** Catalog File with arcs

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "spacecraft",
         "name": "Cassini",
         "startTime": "1997-10-15 09:26:08.390 UTC",
         "arcs": [
            {
               "endTime": "2004-07-01 02:48:00.000 UTC",
               "center": "Sun",
               "trajectory": {
                  "type": "Spice",
                  "target": "Cassini",
                  "center": "Sun"
               },
               "bodyFrame": {
                  "type": "Spice",
                  "name": "CASSINI_SC_COORD"
               }
            },
            {
               "center": "Saturn",
               "trajectory": {
                  "type": "Spice",
                  "target": "Cassini",
                  "center": "Saturn"
               },
               "bodyFrame": {
                  "type": "Spice",
                  "name": "CASSINI_SC_COORD"
               }
            }
         ],
         "geometry": {
            "type": "Mesh",
            "meshRotation": [
               0,
               0,
               -0.70710677,
               0.70710677
            ],
            "size": 0.005,
            "source": "models/cassini/Cassini_no_Huygens_03.3ds"
         },
         "label": {
```

```
            "color": [
               0.6,
               1,
               1
            ]
         },
         "trajectoryPlot": {
            "color": [
               0.6,
               1,
               1
            ],
            "duration": "14 d",
            "fade": 10
         }
      }
   ]
}
```

Example 3: Natural Body Catalog File

Example 3: Saturn and Rings **Natural Body** Catalog File

```
{
   "version": "1.0",
   "name": "Cosmographia CASSINI Example",
   "items": [
      {
         "class": "planet",
         "name": "Saturn",
         "mass": "95.152 Mearth",
         "density": 0.687,
         "center": "Sun",
         "trajectory": {
            "type": "Spice",
            "target": "SATURN",
            "center": "SUN"
         },
         "bodyFrame": {
            "type": "Spice",
            "name": "IAU_SATURN"
         },
         "geometry": {
            "type": "Globe",
            "radii": [
               60268,
               60268,
               54364
            ],
            "baseMap": "textures/saturn.jpg"
         },
         "label": {
            "color": [
               0.8,
               1.0,
               0.5
            ]
         },
         "trajectoryPlot": {
             "duration": "10759.22 d",
             "fade": 0.3
         }
      },
      {
         "name": "Saturn Rings",
         "center": "Saturn",
         "bodyFrame":  {
            "type": "BodyFixed",
            "body": "Saturn"
         },
```

```
            "geometry":
            {
               "type": "Rings",
               "innerRadius": 74660,
               "outerRadius": 140220,
               "texture": "textures/saturn-rings.png"
            }
         }
      ]
}
```